Introducción al Lenguaje Basic

Prof. Luis Gerardo Peña Camacho

El Lenguaje Basic

- **BASIC** es una familia de lenguajes de programación. Fue originalmente ideado como una herramienta de enseñanza, se diseminó entre los microcomputadores caseros en la década de1980, y sigue siendo popular hoy en día en muchos dialectos bastante distintos del original.
- BASIC es el acrónimo de **B**eginners **A**ll-purpose **S**ymbolic **I**nstruction **C**ode (en español: “código de instrucciones simbólicas de propósito general para principiantes”) y está ligado al nombre de un trabajo sin publicar del coinventor del lenguaje, Thomas Kurtz.

Conceptos Básicos
Identificadores

- Los identificadores representan una expresión que hace referencia a una variable o una constantes. Un identificador es una secuencia de caracteres que puede tener una longitud máxima de 255 caracteres.

Conceptos Básicos
Identificadores

Un identificador se caracteriza por estas reglas:

1. Debe comenzar con una letra (A a Z, mayúsculas o minúsculas) y no puede contener blancos.
2. Letras, dígitos y caracteres subrayados (_) están permitidos después del primer carácter.
3. No se puede utilizar una palabra reservada como identificador.
4. El @ Arroba es valido después del último carácter y representa un tipo de datos.
5. El & Ampersand es valido después del último carácter y representa un tipo de datos.

Conceptos Básicos
Identificadores - Ejemplos

Validos

- Nombre
- Area_Rectangulo
- ImpuestoSobreLaRenta
- Peso5

Identifiacdores no Validos

- Area&rectangulo (Tiene un & Ampersand antes del último carácter. Es valido solo al final.)
- Nombre 1 (Tiene un espacio)
- 5apellido (Tiene un número al inicio. Es valido después del primer carácter.)
- For (palabra reservada)

Conceptos Básicos
Palabras Reservadas

- Las palabras reservadas del lenguaje Basic no se pueden utilizar como identificadores, ya que tienen significado especial en Visual Basic y no se utilizan para otros propósitos.

Conceptos Básicos
Palabras Reservadas

- Las palabras reservadas del lenguaje Basic no se pueden utilizar como identificadores, ya que tienen significado especial en Visual Basic y no se utilizan para otros propósitos.

Conceptos Básicos
Palabras Reservadas

Abs	Add	And	App
Activate	Array	Ascatn	Base
Beep	Cbool	Cbyte	Ccur
Cdate	Cdbl	Cdec	Cint
Clng	Csng	Cstr	Cvar
Cverr	Call	Case	Cdh
Choose	Chdir	Chdrive	Clear
Close	Collection	Command	Compare
Const	Cos	CreateObject	Curdir
Dateserial	Datevalue	Day	Ddb
Defftype	Dim	Deletesetting	Dir
Do	Doevents	Each	End
Environ	Eof	Eqv	Erase
Err	Error	Exit	Exp
Explicit	Fileattr	Filecopy	Filedatetime
Filelen	Fix	For	Format
Freefile	Function	Fv	Get
Getattr	GetObject	Getsetting	Getallsetting
Gosub	Goto	Hex	Hour
Imp	Input	Instr	Int
Integer	Ipmt	Irr	Is
Isarray	Isdate	Isempty	Iserror
Ismissing	Isnull	Isnumeric	Isobject
Item	Kill	Lbound	Lcase

Conceptos Básicos
Tipos de Datos

- Los tipos de datos son los distintos objetos de información con los que trabaja una aplicación en Visual Basic. Todos los datos tienen un tipo asociado con ellos. Un dato puede ser un simple carácter como un “B”, una cadena de caracteres como “La casa de pedro”, un valor entero como 242, un número real como 3.1415 o un valor lógico como True o False.

Conceptos Básicos
Clasificación de los Tipos de Datos

- **Enteros**
  - Byte
    - No tiene signo. Rango: 0 a 255. Ideal para almacenar datos binarios
  - Integer
    - 2 bytes. Rango: -32.768 a 32.767. También se declara con el simbolo %
  - Long
    - Entero largo. Rango: -2.147.483.648 y 2.147.483.647. También con el simbolo &
- **Reales**
  - Single
    - punto flotante de precisión simple. 32 bits. Rango: -3,402823E38 a -1,401298E-45. También se declara con el simbolo !
  - Double
    - punto flotante de doble precisión. 64 bits. Rango: -1,79769313486232E308 a -4,94065645841247E-324. También se declara con el simbolo #
  - Currency.
    - 65 bits. Punto fijo con 15 digitos a la derecha y 4 en la parte decimal. Rango: -922.337.203.685.477,5808 a 922.337.203.685.477,5807. También se declara con el simbolo @

Conceptos Básicos
Clasificación de los Tipos de Datos

- **Cadena (String)**
  - Cadena de caracteres. 0 a 65500 caracteres. También se declara con el simbolo $
- **Lógicos**
  - True
  - False
- **Fecha (Date)**
  - Fecha (8 bytes). 1 de enero de 100 a 31 de diciembre de 9999. Indica también la hora, desde 0:00:00 a 23:59:59.
- **Variados (Variant)**
  - Una variable Variant es capaz de almacenar todos los tipos de datos definidos en el sistema. No tiene que convertir entre esos tipos de datos si los asigna a una variable Variant; Visual Basic realiza automáticamente cualquier conversión necesaria.

Conceptos Básicos
Declaración de Constantes

- **Pública (dentro de un módulo)**

  Const nombre_constante = valor

  Public Const PI As Double = 3.1415

  Const Saludo = “Hola”

  Const Max = 1000

- **Privada (en el área de declaraciones general de un formulario)**

  Declarar una *constante privada significa que esa constante puede ser usada dentro de todos los* procedimientos de un mismo formulario o dentro del formulario donde se declara la *constante.*

- **Local (dentro de un** procedimiento)

  Declarar una *constante local significa que esa constante solo puede ser usada dentro del* procedimiento donde se declara.

Conceptos Básicos
Declaración de Variables

- Se utilizan para almacenar temporalmente valores durante la ejecución de la aplicación.
- Declarar una variable consiste en indicarle de antemano al programa el lugar donde se almacenaran los datos desconocidos. Al declarar una variable se debe especificar *el nivel de alcance de la variable, el nombre y el tipo de datos asociado a dicha variable.*

Conceptos Básicos
Declaración de Variables Publicas

- **Public *nombre_variable As Tipo_de_datos***

- Donde:
  - *Public : Indica que la variable es de nivel público.*
  - *nombre_variable: Es un identificador valido para Visual Basic. Este es el nombre que le* permitirá acceder a los valores desconocidos.
  - *As : Palabra clave para indicar el tipo de datos.*
  - *Tipo_de_datos : Indica el tipo de datos que podrá almacenar las variables.* Este corresponde a la clasificación de los tipos de datos.

Conceptos Básicos
Declaración de Variables Privadas

- Dim *nombre_variable* ***As Tipo_de_datos***

  Ejemplos:

  Dim TotalVentasDelDia As Integer

  Dim FacturasImpresa As Integer

  Dim Salario As Single

  Dim Areas as Double, saldo as Single

Conceptos Básicos
Accesibilidad de las Variables

Tipo de variable	Lugar de declaración	Accesibilidad
Global o Public	Declaraciones de *.bas	Desde todos los formularios
Dim o Private	Declaraciones de *.bas	Desde todas las funciones de ese módulo
Public	Declaraciones de *.frm	Desde cualquier procedimiento del propio formulario y desde otros precedida del nombre del modulo en el que se ha declarado
Dim o Private	Declaraciones de *.frm	Desde cualquier procedimiento del propio formulario
Dim	Cualquier procedimiento de un módulo	Desde el propio procedimiento

Conceptos Básicos
Tipos de Operadores

Tipo	Operación	Operador en Vbasic
Aritmético	Exponenciación	^
	Cambio de signo (operador unario)	-
	Multiplicación, división	*, /
	División entera	\
	Resto de una división entera	**Mod**
	Suma y resta	+, -
Concatenación	Concatenar o enlazar	& +
Relacional	Igual a	=
	Distinto	<>
	Menor que / menor o igual que	< <=
	Mayor que / mayor o igual que	> >=
Otros	Comparar dos expresiones de caracteres	**Like**
	Comparar dos referencias a objetos	**Is**
Lógico	Negación	**Not**
	And	**And**
	Or inclusivo	**Or**
	Or exclusivo	**Xor**
	Equivalencia (opuesto a Xor)	**Eqv**
	Implicación (*False* si el primer operando es *True* y el segundo operando es *False*)	**Imp**

Conceptos Básicos
Option Explicit

- Una variable que se utiliza sin haber sido declarada toma por defecto el tipo ***Variant. Puede ocurrir*** que durante la programación, se cometa un error y se escriba mal el nombre de una variable. Por ejemplo, se puede tener una variable " declarada como *entera, y al programar referirse a ella por* error como "; ***Visual Basic supondría que ésta es una nueva variable de tipo Variant***

- Option Explicit, permite declarar previamente las variables que se vayan a usar en la aplicación. En caso de no declararse se mostrara un mensaje de error

Conceptos Básicos
Estructura de Control Selectivas

- Expresiones Lógicas
  - **Simples.** Una unica expresión
    - (A>B)
    - (n<>m)
    - (n+1=m)
  - **Compuestas.** Varias expresiones relacionadas con and, or, not
    - ((a>b)and(c>d))
    - (a=4) or (a=3)

Conceptos Básicos
Estructura de Control Selectivas

- **Operador lógico AND**

  El *operador lógico AND (Y) combina dos o más expresiones lógicas y produce un resultado*

Operando 1		Operando 2	Resultado
True	AND	True	True
True		False	False
False		True	False
False		False	False

Conceptos Básicos
Estructura de Control Selectivas

- **Operador lógico OR**
  - El *operador lógico* ***OR (O) Devuelve verdadero si al menos una de las expresiones lógicas produce*** un resultado verdadero.

Operando 1		Operando 2	Resultado
True	OR	True	True
True		False	True
False		True	True
False		False	False

- **Operador lógico OR**
  - El operador lógico **NOT(NO) niega el valor original de una expresión, si es verdadero será falso, si** es falso será verdadero.

Conceptos Básicos
Estructura de Control Selectivas

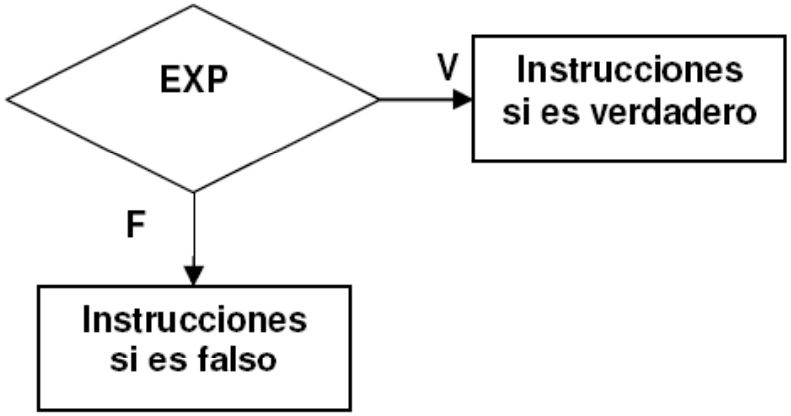
- **Operador lógico OR**
  - El *operador lógico* ***OR (O) Devuelve verdadero si al menos una de las expresiones lógicas produce*** un resultado verdadero.

Operando 1		Operando 2	Resultado
True	OR	True	True
True		False	True
False		True	True
False		False	False

- **Operador lógico OR**
  - El operador lógico **NOT(NO) niega el valor original de una expresión, si es verdadero será falso, si** es falso será verdadero.

Conceptos Básicos
Estructura de Control Selectivas

- **La sentencia If simple**
  - Dado que una condición produce un valor verdadero o falso, se necesita una sentencia de control que ejecute determinada sentencia si la condición es verdadera, y otra si es falsa



Conceptos Básicos
Estructura de Control Selectivas

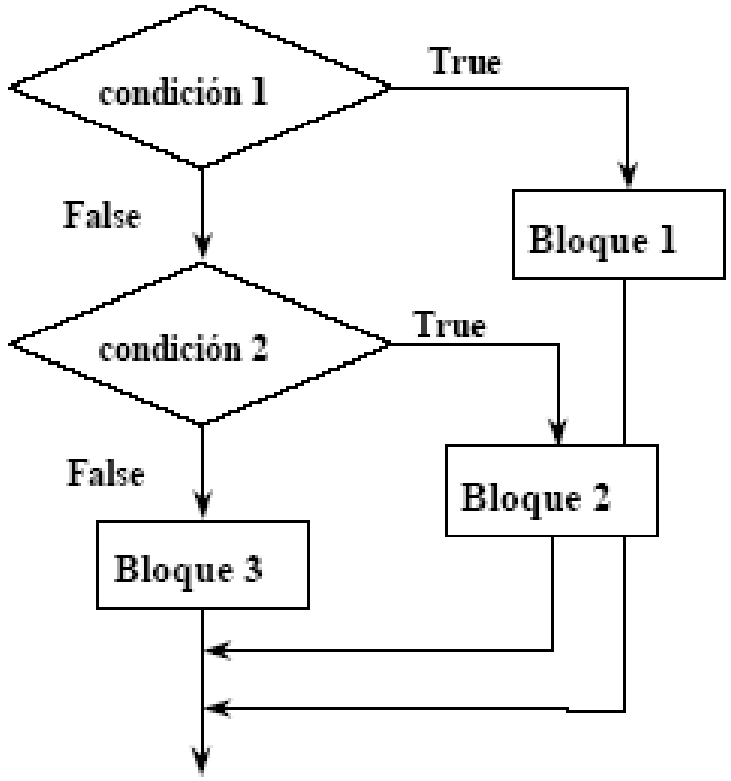
- **La sentencia SI simple. If…Then…Else**

```
If (Condición) Then
[instrucciones si es verdadero]
.
.       .
Else
[instrucciones si es falso]
.       .
.
End If
```

Conceptos Básicos
Estructura de Control Selectivas

- **La sentencia If…Then…Elseif…Then…Else**

```
If condicion1 Then
        sentencias1
ElseIf condicion2 Then
        sentencias2
Else
        sentencia-n
End If
```



Ejemplo:

```
Numero = 53 ’ Se inicializa la va
If Numero < 10 Then
        Digitos = 1
ElseIf Numero < 100 Then
        Digitos = 2
Else
        Digitos = 3
End If
```

Conceptos Básicos
Estructura de Control Selectivas

- **La sentencia Select CASE**

  Esta sentencia permite ejecutar una de entre varias acciones en función del valor de una expresión.

```
Select Case expresion
Case etiq1
        [ sentencias1]
Case etiq2
        [ sentencias2]
Case Else
        sentencias n
End Select
```

Conceptos Básicos
Estructura de Control Selectivas

- **La sentencia Select CASE**

```
Numero = 8
Select Case Numero
Case 1 To 5 ' Numero está entre 1 y 5.
      Resultado = "Se encuentra entre 1 y 5“
Case 6, 7, 8 ' Numero es uno de los tres valores.
      Resultado = "Se encuentra entre 6 y 8"
Case Is = 9 , Is = 10 ' Numero es 9 ó 10.
      Resultado = "El valor es 9 o 10"
Case Else ' Resto de valores.
      Resultado = "El número no se encuentra entre 1 y
                  10"
End Select
```

Conceptos Básicos
Estructura de Control Repetitivas

Las ***Estructuras de Control Repetitivas llamadas también Bucles, son aquellas que nos permiten*** repetir un determinado bloque de código mientras se cumple una determinada condición.

Los *Bucles están compuestos por una condición o expresión que se puede evaluar a falso o verdadero.* Mientras la condición se evalué a verdadero **(true), el cuerpo de bucle se ejecutará.**

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repetir Para. (For…Next)**

  Repite una determinada serie de sentencias dado un valor inicial y un valor final. Este *bucle se utiliza cuando se conoce por anticipado el número de repeticiones* requerida por el programa. Si el número de repeticiones no se conoce por anticipado entonces debe utilizar las sentencia **While … Wend o Do … Loop en lugar de For.**

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repetir Para. (For…Next)**

```
For variable = valor inicial To valor final
        [Sentencias…]
Next variable
```

```
For variable = valor inicial To valor final Step x
        [Sentencias…]
Next variable
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repetir Para. (For…Next)**

```
MyString="Informática "
For Words = 3 To 1 Step -1                ' 3 veces decrementando de 1 en 1.
        For Chars = Words To Words+4      '5 veces.
        MyString = MyString & Chars       ‘ Se añade el número Chars al string.
        Next Chars                        ' Se incrementa el contador
        MyString = MyString & " "         ' Se añade un espacio.
Next Words                                'El valor de MyString es: Informática 34567 23456 12345
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repita Mientras. Do While…Loop**

  repite la ejecución de un conjunto de sentencias mientras una condición dada sea cierta, o hasta que una condición dada sea cierta. La condición puede ser verificada antes o después de ejecutarse el conjunto de sentencias.

Do While *condición*

Instrucciones…

Loop

Hacer Mientras *condición = verdadera*

Instrucciones…

Repite

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repita Mientras. Do While…Loop**

```
' Formato 1:
Do [{While/Until} condicion]
[ sentencias]
[Exit Do]
[ sentencias]
Loop
```

```
' Formato 2:
Do
[ sentencias]
[Exit Do]
[ sentencias]
Loop [{While/Until} condicion]
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repita Mientras. Do While…Loop**

```
Check = True                                ’ Se inicializan las variables.
Counts = 0
Do                                          ' Empieza sin comprobar ninguna condición.
        Do While Counts < 20                 ' Bucle que acaba si Counts>=20 o con Exit Do.
                Counts = Counts + 1   ' Se incrementa Counts.
                If Counts = 10 Then     ' Si Counts es 10.
                        Check = False  ' Se asigna a Check el valor False.
                        Exit Do           ' Se acaba el segundo Do.
                End If
        Loop
Loop Until Check = False                    ' Salir del "loop" si Check es False.
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repita Mientras. Do While…Loop**

```
'Variable para almacenar el total de los caracteres leídos.
Dim TotalCaracterLeido As Integer
'Iniciamos la variable en cero.
TotalCaracterLeido = 0
 'Repetimos mientras la longitud de la caja de texto es mayor que el total de caracteres leído.
Do While Len(txtContenido.Text) > TotalCaracterLeido
        'Incrementamos el total de carácter leído en 1.
        TotalCaracterLeido = TotalCaracterLeido + 1
        'Colocamos el punto de inserción delante del carácter a leer.
        txtContenido.SelStart = TotalCaracterLeido – 1
        'Seleccionamos el carácter.
        txtContenido.SelLength = 1
         ‘Convertimos el carácter seleccionado a mayúscula.
        txtContenido.SelText = UCase(txtContenido.SelText)
'Volvemos a repetir hasta que se lean todos los caracteres de la caja.
Loop
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Bucle Repita Mientras. Do While…Loop**

```
‘Se declara la variable donde se almacenará el número introducido.
Dim Numero As Integer
Do
        ‘Solicita un número y se almacena en la variable numero.
        Numero = InputBox("Introduzca un número:")
        ‘Si el número es negativo se muestra un mensaje al usuario.
        If Numero < 0 Then
                MsgBox ("Introduzca un número positivo.")
'Se repite mientras el número sea negativo.
Loop While (Numero < -0)
'Cuando el numero es positivo el bucle finaliza y se muestra la raíz.
MsgBox ("La raíz del número es:" & Sqr(Numero))
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Sentencia WHILE…WEND**

  Esta sentencia es otra forma de generar bucles que se recorren mientras se cumpla la condición inicial. Su estructura es la siguiente:

Por ejemplo,

```
Counts = 0                    ’ Se inicializa la variable.
While Counts < 20             ’ Se comprueba el valor de Counts.
      Counts = Counts + 1     ’ Se incrementa el valor de Counts.
Wend
```

Conceptos Básicos
Estructura de Control Repetitivas

- **Sentencia FOR EACH ... NEXT**

Esta construcción es similar al bucle ***For, con la diferencia de que la variable que controla la*** repetición del bucle no toma valores entre un mínimo y un máximo, sino a partir de los elementos de un array (o de una colección de objetos).

For Each ***elemento In grupo***
Instrucciones…
[Exit For]
Instrucciones…
Next ***[elemento]***

```
Dim Nombres As New Collection    'Creamos el objeto.
Dim Leido As Variant             'Variable donde se almacenarán los elementos leídos.

'Agregamos datos a la colección.
Nombres.Add "Carlos"
Nombres.Add "Pablo"
Nombres.Add "Jose"

'Leemos cada uno de los elementos agregados.
For Each Leido In Nombres
   MsgBox (Leido)       'Mostramos los elementos leídos en una caja de mensaje.
Next
```

Funciones Pre-Definidas en Visual Basic

Aritméticas	
Formato	**Descripción**
Round(*Expresión.decimal)* Round(5.5)) 'Devuelve 6.	Se utiliza para redondear un número, es decir, devuelve el entero más próximo al argumento.
Int (*número)* MiNumero = **Int(99.8) ' Devuelve 99.**	devuelve el primer número entero negativo menor o igual que el número;
Fix (*número)* MiNumero = **Fix(-99.8) ' Devuelve -99.**	devuelve el primer entero negativo mayor o igual que el número.
Abs (*número)* MiNumero = **Abs(-5) 'Devuelve 5.**	Devuelve el valor absoluto de un número, es decir, el mismo numero si es positivo, o su opuesto, si es negativo.
Exp(*número)* MiNumero = Exp(1) 'Devuelve 2.71828182845905. Es equivalente al valor de la constante e. MiNumero = Exp(2) 'Devuelve 7.38905609893065. Es equivalente al cuadrado de la constante.	Devuelve un tipo **Double que especifica *e (la base de los logaritmos naturales) elevado a una*** potencia. El valor de la constante ***e es 2.718282 aproximadamente.***

Funciones Pre-Definidas en Visual Basic

Aritméticas	
Formato	**Descripción**
Log(*número)* MiLogaritmo = **Log(3) 'Devuelve 1.09861228866811.**	Devuelve un tipo **Double que representa el logaritmo natural de un número.**
Sqr(*número)* Raiz = **Sqr(25) 'Devuelve 5.** Raiz = **Sqr(4) 'Devuelve 2.**	Devuelve la raíz cuadrada de un número.
Sgn(*número)* Signo = **Sgn(-5) 'Devuelve -1.** Signo = **Sgn(4) 'Devuelve 1.** Signo = **Sgn(0) 'Devuelve 0.**	Devuelve un tipo **Integer que indica el signo de un número.**

Funciones Pre-Definidas en Visual Basic

Aritméticas	
Formato	**Descripción**
Sin(*número)* MiSeno = **Sin(1.4)**	Devuelve un tipo **Double que especifica el seno de un ángulo expresado en** radianes
Cos(*número)* MiCoseno = **Cos(3.5)**	Devuelve el coseno de un ángulo expresado en radianes.
Tan(*número)*	Devuelve la tangente de un ángulo expresado en radianes.
Atn(*número)*	Devuelve la arcotangente de un ángulo expresado en radianes.

Funciones Pre-Definidas en Visual Basic – funciones derivadas

Función	Derivadas equivalentes
Secante	Sec(X) = 1 / Cos(X)
Cosecante	Cosec(X) = 1 / Sin(X)
Cotangente	Cotan(X) = 1 / Tan(X)
Seno inverso	Arcsin(X) = Atn(X / Sqr(-X * X + 1))
Coseno inverso	Arccos(X) = Atn(-X / Sqr(-X * X + 1)) + 2 * Atn(1)
Secante inversa	Arcsec(X) = Atn(X / Sqr(X * X – 1)) + Sgn((X) – 1) * (2 * Atn(1))
Cosecante inversa	Arccosec(X) = Atn(X / Sqr(X * X – 1)) + (Sgn(X) – 1) * (2 * Atn(1))
Cotangente inversa	Arccotan(X) = Atn(X) + 2 * Atn(1)
Seno hiperbólico	HSin(X) = (Exp(X) – Exp(–X)) / 2
Coseno hiperbólico	HCos(X) = (Exp(X) + Exp(–X)) / 2
Tangente hiperbólica	HTan(X) = (Exp(X) – Exp(–X)) / (Exp(X) + Exp(–X))
Secante hiperbólica	HSec(X) = 2 / (Exp(X) + Exp(–X))
Cosecante hiperbólica	HCosec(X) = 2 / (Exp(X) – Exp(–X))
Cotangente hiperbólica	HCotan(X) = (Exp(X) + Exp(–X)) / (Exp(X) – Exp(–X))
Seno hiperbólico inverso	HArcsin(X) = Log(X + Sqr(X * X + 1))
Coseno hiperbólico inverso	HArccos(X) = Log(X + Sqr(X * X – 1))
Tangente hiperbólica inversa	HArctan(X) = Log((1 + X) / (1 – X)) / 2
Secante hiperbólica inversa	HArcsec(X) = Log((Sqr(–X * X + 1) + 1) / X)
Cosecante hiperbólica inversa	HArccosec(X) = Log((Sgn(X) * Sqr(X * X + 1) + 1) / X)
Cotangente hiperbólica inversa	HArccotan(X) = Log((X + 1) / (X – 1)) / 2
Logaritmo en base N	LogN(X) = Log(X) / Log(N)

Funciones de Conversión de Tipos Visual Basic

- En el siguiente ejemplo se muestra como calcular el área de un triangulo donde la base y la altura son proporcionadas mediante cajas de texto.

```
Dim Area, Base, Altura As Double
'Leemos la base de la caja de texto txtBase.
Base = CDbl(txtBase.Text)
'Leemos la altura de la caja de texto txtAltura.
Altura = CDbl(txtAltura.text)
'Calculamos el área del triángulo.
Area = (Base*Altura)/2
'Muestra el área en un cuadro de mensaje.
MsgBox(Area)
```

Mediante la función **CDbl podemos convertir el tipo de datos String (cadena) devuelto por la caja** de texto al tipo de datos numérico **Double.**

Funciones de Conversión de Tipos Visual Basic

Función	Tipo devuelto	Intervalo del argumento *expresión*
Cbool	Boolean	Cualquier expresión de cadena o numérica válida.
Cbyte	Byte	0 a 255.
Ccur	Currency	-922.337.203.685.477,5808 a 922.337.203.685.477,5807.
Cdate	Date	Cualquier expresión de fecha.
CDbl	Double	-1,79769313486232E308 a -4,94065645841247E-324 para valores negativos; 4,94065645841247E-324 a 1,79769313486232E308 para valores positivos.
Cdec	Decimal	+/-79.228.162.514.264.337.593.543.950.335 para números basados en cero, es decir, números sin decimales. Para números con 28 decimales, el intervalo es +/-7,9228162514264337593543950335. La menor posición para un número que no sea cero es 0,0000000000000000000000000001.
CInt	Integer	-32.768 a 32.767; las fracciones se redondean.
CLng	Long	-2.147.483.648 a 2.147.483.647; las fracciones se redondean.
CSng	Single	-3,402823E38 a -1,401298E-45 para valores negativos; 1,401298E-45 a 3,402823E38 para valores positivos.
CStr	String	El mismo intervalo que Double para valores numéricos. El mismo intervalo que String para valores no numéricos.
Cvar	Variant	El valor de retorno de CStr depende del argumento *expresión*.

Que estudiamos del Lenguaje Basic

- Identificadores
- Palabras reservadas
- Tipos de Datos: Enteros, reales, cadenas, logicos y variados
- Constantes: Declaración, públicas, privadas y locales
- Variables
- Estructuras Selectivas
  - Expresiones Lógicas: aritméticos, relacionales y lógicos
  - La sentencia If y la sentencia CASE
- Estructuras Repetitivas
  - Bucle For…Next (Repita Para)
  - Bucle Do…Loop (Repita Mientas)
  - Bucle While….Wend (Mientras se cumpla, parecido al For)
  - Buble For Each…Next (parecido al for, con repeticiones en un array)
- Funciones pre-definidad
- Conversiones de tipo

Lectura complementaria

- Lectura de los capítulos 2, 3 y 4 del libro guía “Introducción a la Programación con Aplicaciones en Visual Basic” de la Profa. Maria Alejandra Quintero . Descargar aquí

- Lectura del capítulo 3. Lenguaje Basic del libro guia “Aprenda Visual Basic como si estuviera en primero” Descargar aquí

- Fecha del próximo parcial Martes 7 de Junio de 2011